WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Friday, July 09, 2004

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ	
☐	L9	L5 and "power of 2"	0
☐	L8	L5 and I2	19
☐	L7	L6 and I2	0
		DB=EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L6	L5	9
		DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ	
☐	L5	L4 and (second near2 (counter or address indicator))	438
☐	L4	L3 and (packet or instruction unit or execution unit)	4960
☐	L3	(counter or address indicator) same instruction	29419
		DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ	
☐	L2	L1 or 712/208-213,230-232.ccls.	2311
☐	L1	717/127-131.ccls.	927

END OF SEARCH HISTORY

Terms used **instruction packet** AND **counter**   Found **6,504** of **139,567**

Sort results by   relevance ▾   ❖ Save results to a Binder
Display results   expanded form ▾   ⬚ Search Tips
                                               ☐ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200   Result page: **1** 2 3 4 5 6 7 8 9 10   next
Best 200 shown   Relevance scale ☐ ▬ ▪ ▪ ▪

1 Embedded applications: AES and the cryptonite crypto processor   ▪
Dino Oliva, Rainer Buchty, Nevin Heintze
October 2003 **Proceedings of the international conference on Compilers, architectures and synthesis for embedded systems**

Full text available: 📄 pdf(346.09 KB)   Additional Information: full citation, abstract, references, index terms

CRYPTONITE is a programmable processor tailored to the needs of crypto algorithms. The design of CRYPTONITE was based on an in-depth application analysis in which standard crypto algorithms (AES, DES, MD5, SHA-1, etc) were distilled down to their core functionality. We describe this methodology and use AES as a central example. Starting with a functional description of AES, we give a high level account of how to implement AES efficiently in hardware, and present several novel optimizations (whic ...

Keywords: AES, architecture, cryptography, high-bandwidth, high-speed, processor, round key generation, software implementation

2 Automatic formal verification for scheduled VLIW code   ▪
Xiushan Feng, Alan J. Hu
June 2002 **ACM SIGPLAN Notices , Proceedings of the joint conference on Languages, compilers and tools for embedded systems: software and compilers for embedded systems**, Volume 37 Issue 7

Full text available: 📄 pdf(113.92 KB)   Additional Information: full citation, abstract, references, index terms

VLIW processors are attractive for many embedded applications, but VLIW code scheduling, whether by hand or by compiler, is extremely challenging. In this paper, we extend previous work on automated verification of low-level software to handle the complexity of modern, aggressive VLIW designs, e.g., the exposed parallelism, pipelining, and resource constraints. We implement these ideas into a prototype tool for verifying short sequences of assembly code for TI's C62x family of VLIW DSPs, and dem ...

Keywords: DSP, VLIW, formal verification, symbolic execution, theory of equality with uninterpreted functions

3 Architecture and implementation of a VLIW supercomputer   ▪
Robert P. Colwell, W. Eric Hall, Chandra S. Joshi, David B. Papworth, Paul K. Rodman, James E. Tornes

November 1990 **Proceedings of the 1990 ACM/IEEE conference on Supercomputing**

Full text available: pdf(1.29 MB)   Additional Information: full citation, abstract, references

Very-Long-Instruction-Word (VLIW) computers achieve high performance by exploiting the fine-grain parallelism present in sequential or vectorizable code. Multiflow's /200 and /300 VLIW systems yielded near-supercomputer performance by this means despite the relatively slow (65 nS) clocks. With its much faster clock period (15 nS) and architectural improvements, the new /500 system attains approximately 4-9X the performance of its predecessors.This paper describes the /500 architecture and implem ...

4 Session 4: Optimal organizations for pipelined hierarchical memories

Gianfranco Bilardi, Kattamuri Ekanadham, Pratap Pattnaik

August 2002 **Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures**

Full text available: pdf(229.84 KB)   Additional Information: full citation, abstract, references, index terms

In a recent paper (SPAA'01), we have established that the Pipelined Hierarchical Random Access Machine (PH-RAM) is a powerful model of computation, where most of the memory latency can be hidden by concurrency of accesses. In the present work, we explore the physical feasibility of PH-RAMs.A pipelined hierarchical memory of size S is characterized by two metrics: the access function $a(\&khgr;)$, denoting the time required by an access to location x, and the pipeline period $p(S)$, denoti ...

Keywords: hierarchical memory processor, scalable pipeline

5 Memory hierarchies: A code decompression architecture for VLIW processors

Yuan Xie, Wayne Wolf, Haris Lekatsas

December 2001 **Proceedings of the 34th annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.00 MB)   Additional Information: full citation, abstract, references, citings
Publisher Site

In embedded system design, memory has been one of the most restricted resources. Reducing program size has been an important goal when designing an embedded system. Most of the previous work on code compression has targeted RISC architectures. Recently VLIW processors became very popular, particularly for signal processing. Decompression speed is especially important for VLIW architectures given that the length of the instruction word is long. Furthermore, modern VLIW architectures use flexible ...

6 A multi-user data flow architecture

F. J. Burkowski

May 1981 **Proceedings of the 8th annual symposium on Computer Architecture**

Full text available: pdf(606.85 KB)   Additional Information: full citation, abstract, references, citings, index terms

This paper discusses the design of a prototype data flow machine that has memory management hardware in each memory block. This facility allows loading and deleting code that is produced by independent compilations. The first sections of the paper deal with the general architecture of the machine and the format specifications for the instruction cells, logical addresses, and switch packets. The paper concludes with a discussion of the mapping hardware used in the memory blocks. The results ...

7 Data-Driven and Demand-Driven Computer Architecture

Philip C. Treleaven, David R. Brownbridge, Richard P. Hopkins

January 1982 **ACM Computing Surveys (CSUR)**, Volume 14 Issue 1

Full text available: pdf(4.14 MB)　　Additional Information: <u>full citation</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

8 <u>Database machines: Design considerations for data-flow database machines</u>
Haran Boral, David J. DeWitt
May 1980　**Proceedings of the 1980 ACM SIGMOD international conference on Management of data**
Full text available: pdf(1.20 MB)　　Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>

This paper presents a discussion of the application of data-flow machine concepts to the design and implementation of database machines which execute relational algebra queries. We analyze the performance of multiprocessor nested-loops and sort-merge join algorithms and show that the nested-loops algorithm is generally superior. Three levels of operand granularity for data-flow database machines are introduced and compared using the nested-loops join algorithm. We demonstrate, that relation-leve ...

9 <u>Application specific compiler/architecture codesign: a case study</u>
Oliver Wahlen, Tilman Glökler, Achim Nohl, Andreas Hoffmann, Rainer Leupers, Heinrich Meyr
June 2002　**ACM SIGPLAN Notices , Proceedings of the joint conference on Languages, compilers and tools for embedded systems: software and compilers for embedded systems**, Volume 37 Issue 7
Full text available: pdf(290.01 KB)　　Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>index terms</u>

This paper proposes an architecture exploration methodology for application specific instruction set processors (ASIPs) including a C compiler and a VHDL model in the exploration loop. For a given application the target architecture is an instance of the scalable ALICE VLIW architecture which will be presented in this paper. In a case study it will be explained how the LISA processor design platform in conjunction with the CoSy compiler environment significantly reduces the time for exploration ...

Keywords: ASIP, architecture exploration, retargetable compiler

10 <u>Hardware acceleration of logic simulation using a data flow microarchitecture</u>
G. Catlin, B. Paseman
December 1985　**ACM SIGMICRO Newsletter , Proceedings of the 18th annual workshop on Microprogramming**, Volume 16 Issue 4
Full text available: pdf(733.15 KB)　　Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>index terms</u>

Current digital logic simulators running on engineering workstations lack capacity and speed. This paper discusses a hardware accelerator for a workstation simulator which addresses these problems. The accelerator runs 100x faster than its software counterpart and can simulate up to 1 million gates. The accelerator has been built and is being sold commercially. The architecture of the accelerator is similar to that of a classical dataflow machine. We describe the architecture of the machine ...

11 <u>A bidirectional data driven Lisp engine for the direct execution of Lisp in parallel</u>
C. K. Yuen, W. F. Wong
June 1989　**ACM SIGARCH Computer Architecture News**, Volume 17 Issue 4
Full text available: pdf(761.13 KB)　　Additional Information: <u>full citation</u>, <u>index terms</u>

12 <u>A data flow architecture with a paged memory system</u>
L. J. Caluwaerts, J. Debacker, J. A. Peperstraete
April 1982　**Proceedings of the 9th annual symposium on Computer Architecture**

Full text available: pdf(549.31 KB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

During the last ten years, data flow has become an exciting research area and several architectures have been proposed and built. They differ mostly in the way they handle data structures and how they provide mechanisms for token labeling or colouring in order to make data flow graphs reentrant. The paper presents a data flow architecture with a paged memory system to hold both data flow programs and data structures. The token labeling mechanism is coupled with the memory managem ...

Keywords: Data flow, Paged memory, Parallel processing

13 <u>The complexity of the equivalence problem for counter machines, semilinear sets, and simple programs</u>

Eitan M. Gurari, Oscar H. Ibarra
April 1979 **Proceedings of the eleventh annual ACM symposium on Theory of computing**

Full text available: pdf(931.25 KB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>index terms</u>

It is shown that the class of relations (functions) definable by Presburger formulas is exactly the class of relations (functions) computable by finite-reversal multicounter machines. An upper bound of $2c(N/logN)4$ on the deterministic time complexity of the equivalence problem for such machines is established. In fact, it is proved that the inequivalence problem is NP-complete. These results are used to derive some upper bounds on the com ...

14 <u>Performance analysis using the MIPS R10000 performance counters</u>

Marco Zagha, Brond Larson, Steve Turner, Marty Itzkowitz
November 1996 **Proceedings of the 1996 ACM/IEEE conference on Supercomputing (CDROM)**

Full text available: pdf(200.39 KB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

Tuning supercomputer application performance often requires analyzing the interaction of the application and the underlying architecture. In this paper, we describe support in the MIPS R10000 for non-intrusively monitoring a variety of processor events -- support that is particularly useful for characterizing the dynamic behavior of multi-level memory hierarchies, hardware-based cache coherence, and speculative execution. We first explain how performance data is collected using an integrate ...

15 <u>Minimal shift counters and frequency division</u>

Alice M. Tokarnia
July 1993 **Proceedings of the 30th international conference on Design automation**

Full text available: pdf(603.15 KB) Additional Information: <u>full citation</u>, <u>references</u>, <u>index terms</u>

16 <u>Flowchart schemata with counters</u>

David A. Plaisted
May 1972 **Proceedings of the fourth annual ACM symposium on Theory of computing**

Full text available: pdf(1.01 MB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

The translation of a specific flowchart schema with one counter into an equivalent flowchart schema without counters is described. This result leads easily to the general translation

method from one-counter flowchart schemata to zero-counter flowchart schemata. Some generalizations are then presented.

17 Reversal complexity of counter machines

Tat-hung Chan

May 1981 **Proceedings of the thirteenth annual ACM symposium on Theory of computing**

Full text available: pdf(1.05 MB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>index terms</u>

It has long been known that deterministic 1-way counter machines recognize exactly all r.e. sets. Here we investigate counter machines with general recursive bounds on counter reversals. Our main result is that for bounds which are at least linear, counter reversal is polynomially related to Turing machine time, for both 1-way and 2-way counter machines and in both the deterministic and the nondeterministic cases. This leads to natural characterizations of the classes P and NP, and hence of ...

18 Efficient implementation of a statistics counter architecture

Sriram Ramabhadran, George Varghese

June 2003 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems**, Volume 31 Issue 1

Full text available: pdf(276.19 KB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>index terms</u>

Internet routers and switches need to maintain millions of (e.g., per prefix) counters at up to OC-768 speeds that are essential for traffic engineering. Unfortunately, the speed requirements require the use of large amounts of expensive SRAM memory. Shah et al [1] introduced a cheaper statistics counter architecture that uses a much smaller amount of SRAM by using the SRAM as a cache together with a (cheap) backing DRAM that stores the complete counters. Counters in SRAM are periodically updated ...

Keywords: router, statistics counter

19 Synchronous up/down binary counter for LUT FPGAs with counting frequency independent of counter size

Alexandre F. Tenca, Miloš D. Ercegovac

February 1997 **Proceedings of the 1997 ACM fifth international symposium on Field-programmable gate arrays**

Full text available: pdf(563.06 KB)   Additional Information: <u>full citation</u>, <u>references</u>, <u>index terms</u>

20 Self-stabilization by counter flushing

George Varghese

August 1994 **Proceedings of the thirteenth annual ACM symposium on Principles of distributed computing**

Full text available: pdf(1.02 MB)   Additional Information: <u>full citation</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

Results 1 - 20 of 200   Result page: **1** <u>2</u> <u>3</u> <u>4</u> <u>5</u> <u>6</u> <u>7</u> <u>8</u> <u>9</u> <u>10</u>   <u>next</u>

BEST AVAILABLE COPY

CiteSeer Find: | instruction packet AND counter | [Documents] [Citations]

Searching for **instruction packet and counter**.
Restrict to: <u>Header</u> <u>Title</u> Order by: <u>Expected citations</u> <u>Hubs</u> <u>Usage</u> <u>Date</u> Try: <u>Google (CiteSeer)</u> <u>Google</u> <u>(Web)</u> <u>CSB</u> <u>DBLP</u>
2 documents found. **Order: number of citations.**

<u>IDIAS Dataflow Machine Simulator - Roh (1992)</u> <u>(Correct)</u> <u>(2 citations)</u>
When there is a match, it builds a ready **instruction packet** and sends it to the ready queue. ffl The
Model In a von Neumann architecture, the program **counter** dictates the sequence of instructions to be
www.cs.colostate.edu/~dataflow/html/../papers/Manuals/id.ps.gz

<u>A Replay Mechanism for Massively Parallel Computer RWC-1 - Nobuyuki Ichiyoshi (1994)</u> <u>(Correct)</u>
is sent out into the network by a make **packet instruction**, and a message arrival automatically
basic replay scheme that uses software instruction **counter** for relatively low-overhead recording of timings
jisp.cs.nyu.edu/RWC/rwcp/papers/1994/B-21_183.ps.gz

Try your query at: <u>Google (CiteSeer)</u> <u>Google (Web)</u> <u>CSB</u> <u>DBLP</u>

CiteSeer - Copyright <u>NEC</u> and <u>IST</u>